DOCUMENT RESUME

ED 388 300 IR 017 449

AUTHOR Vadaparty, K.; And Others
TITLE On the Design and Development of Pedagogy-First CAI Tools for CS Education.
PUB DATE 94
NOTE 7p.; In: Educational Multimedia and Hypermedia, 1994. Proceedings of ED-MEDIA 94--World Conference on Educational Multimedia and Hypermedia (Vancouver, British Columbia, Canada, June 25-30, 1994); see IR 017 359.
PUB TYPE Reports - Research/Technical (143) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Algorithms; Animation; *Computer Assisted Instruction; *Computer Science Education; *Computer Software Development; Data; Graduate Study; Higher Education; *Instructional Design; *Multimedia Materials; Nontraditional Education; Undergraduate Study

ABSTRACT

This paper presents the implications of an ongoing project on the design and development of multimedia instructional material for teaching and learning computer science topics at both graduate and undergraduate levels. Important pedagogical requirements that CAI software should satisfy include: (1) animation of the changes in tree topologies; (2) coordination of the text of the algorithm, the explanation of what is happening at the current step and the interactive animation; (3) superimposition of meta-algorithmic information; (4) presentation of dynamic examples; (5) the ability to create dynamic quizzes; and (6) augmenting data structures with procedures. Subject matter is divided into themes, as in traditional classroom instruction, and the software is developed for each of these themes. The current theme is tree data structures, such as binary trees, red-black trees, and 2-3 trees. The tools are intended to relieve students of the burden of visualizing the data structures and to enable teachers to concentrate more on the principles behind the data structures than on the mechanical operations. (Contains 10 references.) (AEF)

On the design and development of Pedagogy-First CAI tools for CS Education

K. Vadaparty H. Salem W. Thompson S. Mathew
Department of Computer Engineering and Science,
Case Western Reserve University, Cleveland, OH 44106

R. Molmen
Baldwin-Wallace College
Berea, OH 44017

Abstract: In this paper we present the results of an ongoing project on the *design and development* of multimedia instructional material for teaching and learning a number of computer science topics at both graduate and undergraduate levels. We emphasize on pedagogical requirements such as *dynamic* animation of the changes in the data structures induced by the user-input, *dynamic* automated examinations that are *not* multiple-choice types, incorporation of meta-algorithmic information, the ability to step through algorithms both forward and backward, etc. We divide the subject-matter into *themes* and develop the software for each of these themes. The current theme we considered is tree data structures such as binary trees, red-black trees, 2-3 trees, etc. We believe that our tools relieve the students of the burden of visualizing the data structures and also enable the teachers to concentrate more on the principles behind the data structures than on the mechanical operations. We believe that our project is one of the first to exploit the sophisticated PC tools for CAI.

1 Overview

In this paper we present the results of an ongoing project on the *design and development* of multimedia instructional material for teaching and learning a number of computer science topics at both graduate and undergraduate levels. One of our *fundamental goals* in this project is in developing computer aided instruction software *that meets a number of important pedagogical* requirements. We believe that the progress of the project so far, as reported in this paper, *makes a significant first step in achieving this goal.* There are a few questions one needs to ask regarding CAI tools. First, is there a need for more CAI tools. Our answer is a definite "yes". In brief, our position on these systems is the same as expressed in [NVH86]:

"During the 1970's a second wave of CAI systems [started]. Despite significant government and commercial commitments to large-scale CAI projects, the record of actual use has been spotty ... recent proliferation of "smart" machines capable of instructional ... [with graphics capabilities] ... and above all, affordable prices... has opened new areas in which ... [it is possible to operate with] "learning-by-doing" paradigm... These ad hoc, "do-it-yourself" uses of CAI techniques will ultimately succeed where frontal attacks have failed ..."

Powerful hardware and software tools in the PC world have never been more affordable than now. We believe that our project will start new developments that hopefully exploit the power of modern PCs

in computer science education.

The second question is who should develop CAI tools: since these tools should "distill" the experience of the teachers (and the difficulties of students), we strongly believe that CAI tools (prototypes at any rate) should be developed by teachers and students, although they may be marketed by publishers or the software industry (perhaps after adding more features). Finally, the third question is what should the level of genericity be. We believe that computer aided instruction ought to divide the subject matter into *themes*, just as we do in classroom instruction. A theme is a family of lessons that teach a particular algorithmic or structural paradigm. Examples of themes abound: *tree data structures*, *graph search algorithms*, *sorting*, *sweepline paradigm*, *divide and conquer paradigm*, etc.

The main contribution of our work is in *designing and developing* computer aided instruction software for the tree data structures *theme*: it satisfies a number of important pedagogical requirements as described in Section 2. Furthermore, it is *readily usable* by teachers (students) for teaching (learning) tree data structures. A student or teacher can run (client configuration) our system on MS-Windows at almost minimal configurations.

The paper is organized as follows. Section 2 describes the desired pedagogical requirements that any CAI system should satisfy; most of these requirements are satisfied by our system. Section 3 briefly surveys the related work and provides a comparison with some of the major CAI efforts. The paper concludes with screen-captures of actual running examples.

Implementation: The project, implemented in Visual Basic 3.0, has about 32,000 lines of code and took about 3.5 person-years.

2 Pedagogical requirements that our system satisfies

In this section we describe some of the important pedagogical requirements that any CAI software for computer science lessons should satisfy. Note that our software satisfies most of these as shown in the illustrations later. Some of the issues are specific to the *theme* of tree data structures while some may be useful for other themes as well. To the best of our knowledge, many of these requirements are not met by any of the CAI software currently available; textbooks certainly do not come even close to meeting these. Most of these requirements are currently met by the efforts of enthusiastic instructors, teaching assistants, and sometimes students themselves. The following discussion refers to a number of tree data structures which can be found in classical references such as [CLR90, Tar83]. Our listing here is not intended to be exhaustive and exclusive; it is the best we could get from our own experience and the experience of others [Nie90, NVH86, EH92, Fla87, Sto89].

1. *Interactive Animation*: When using tree data structures, one of the most important issues is the change in the topology of the tree when a new value is inserted or deleted. This may involve one or more rotations (AVL trees, red-black trees, splay trees), splitting (2-3 trees, B-trees, etc), change of attribute-values of the tree-nodes (change of color in red-black trees, values in order-statistics trees), and so on. Any software purported to aid the instruction about these tree data structures should animate the changes of the tree topologies. Furthermore, such an animation should be *user directed*.

2. *Stepping Through*: It is of great help to the student see how the updates algorithms (delete/insert) modify the tree data structures step by step. This involves coordinating the text of the algorithm, the explanation of what is happening at the current step and the interactive animation of the tree while the current step is being executed.

3. *Superimposition of Meta-algorithmic Information*: When a tree topology changes due to updates, it follows certain rules. These rules are specified through *canonical cases*. For instance, in the case of red-black trees, [CLR90] describes three canonical cases for insertion into red-black trees. A user-defined example must show which canonical case that it is using to perform the current rotation, etc. Note that, strictly speaking, this superimposition is not a part of the algorithm; it is "meta-algorithmic" information.

4. *Pathological Examples* that are dynamic: We observed in our teaching and learning that it is always good to present "killer examples", i.e., examples that are absolutely bad for the particular algorithm or data structure. It is, for example, highly instructive to show an order of insertions which causes a skewed binary search tree and then show how this very order of insertions produces a balanced red-black tree. Furthermore, these examples should be dynamic, that is, different at different runs of the software.

5. *Dynamic Quizzes*: An important feature of learning process is also "doing". We believe that one way to help the students participate in the understanding process is to make them answer questions. The system should be able to "create" dynamic quizzes. Note that these should not be just multiple-choice questions, but should require user participation that is more involving: say, (i) insert/delete (ii) perform the consequent fixup steps (rotations) in a red-black tree, etc. This type of user participation increases understanding of the subject matter better [Sto89, Fla87].

6. *Augmenting Data Structures* with procedures: One should have procedures added to the nodes of a data structure to evaluate certain properties. For instance, each node of a tree can be augmented with a procedure that computes the *predecessor* (or *successor*) of the node.

3 Related Work and a Comparison

Although there is an abundance of software for teaching concepts at the K-12 level of education, there are very few popular software packages for teaching Computer Science lessons at the university level. An excellent description of ups and downs of CAI from sixties through eighties is presented in [NVH86]. Some of the examples of CAI tools are PLATO whose derivative was AUTOTOOL[Sto89], and BALSA and ZEUS [BS84, Bro92]. In principle, we could have used Zeus to develop lessons on various themes (as is done for red-black trees [Hey93]). However, we opted not to use Zeus for a number of reasons. We observed that considerable amount of work needs to be done to develop teaching tools that can be readily used. For example, labeling nodes, indicating canonical cases, etc., requires special view development and coding into the annotations. We believe that using sophisticated software packages and inexpensive graphics hardware available for PC's one can develop the desired instructional software from scratch with the same or less effort. This belief is bolstered by our development of a number of data structures. Note that Zeus is a general purpose algorithm animating environment, and was not developed only for educational tools; unlike Zeus, our software was developed for educational purposes only, and can not be used to animate arbitrary algorithms.

Acknowledgements: Kumar Vadaparty would like to acknowledge the partial support of the Lilly Foundation for the Fellowship award which included a semester's release time. He also would like to thank Dr. Ray Neff, Dr. Sandra Russ, Dr. Tom Kicher and Dr. Lee White. As a team, we would also like to thank Mr. Jamie Carl, for the initial work of developing binary search tree animation using Turbo Pascal. Thanks go to Ms. Dawn Cohen for pointing out the idea of dynamic quizzes.

References

[Bro92] Brown, Marc H. Zeus: A system for algorithm animation and multi-view editing. SRC Research Report 75, DEC, February 1992.

[BS84] Brown, Marc and Segewick, Robert. A system for algorithm animation. *Computer Graphics*, 18(3):177–186, 1984.

[CLR90] Cormen, T., Leiserson, C., and Rivest, R. *Introduction to Algorithms*, volume 1 of *Computer Science*. McGraw Hill, 1990.

[EH92] Edwards, D.N.A and Holland, S. *Multimedia Interface Design in Education*, volume 76 of *Computer and System Sciences*. Springer-Verlag, 1992.

[Fla87] Flanders, H. Mathematical Calc. Education. In *AMS (summer) Minicourse, Salt Lake City, Utah*, 1987.

[Hey93] Heydon, Allan. Search tree animation. Technical report, DEC, 1993.

[Nie90] Nievergelt, J. Computer Science for Teachers: A Quest for Classics and How to Present Them. In *Proceedings of the second international conference on computer assisted learning*, 1990.

[NVH86] Nievergelt, J, Ventura, A., and Hinterberger, H. *Interactive Computer Programs for Education-Philosophy, Techniques, and Examples*. Addison-Wesley, 1986.

[Sto89] Stone, M.G. Interactive COSTOC Tutorials. In *Proceedings of the second international conference on computer assisted learning*, 1989.

[Tar83] Tarjan, R.E. *Data Structures and Network Algorithms*, volume 1 of *Applied Mathematics*. CBMS-NSF, 1983.

Examples

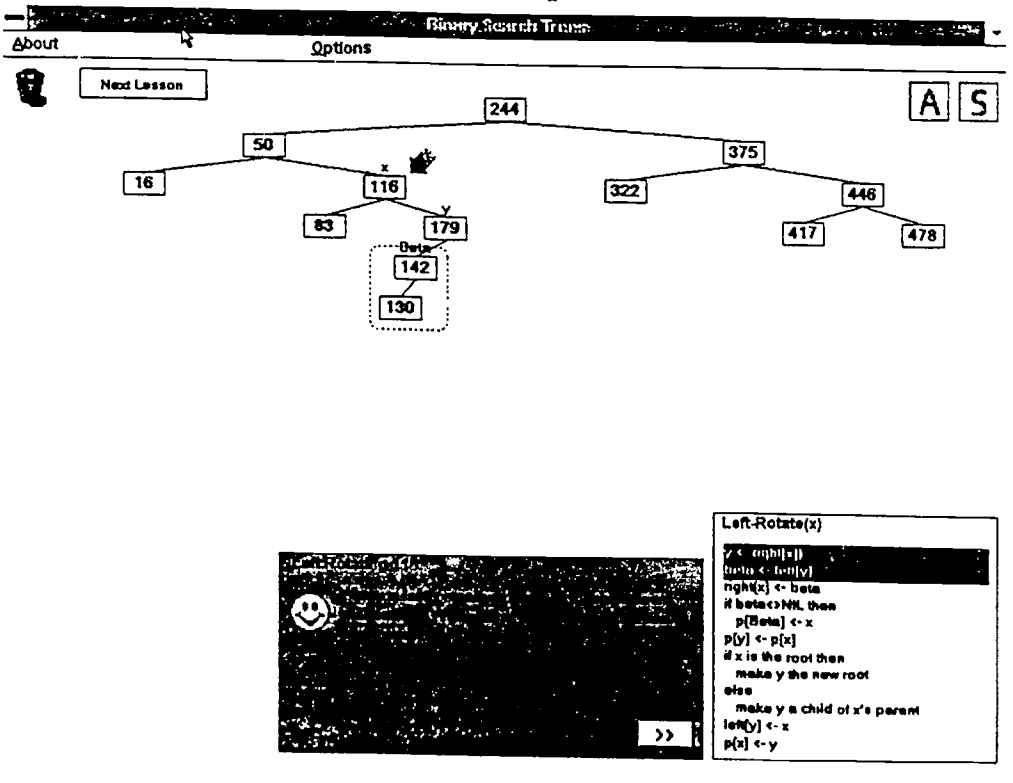

Figure 1. An example of a Binary Tree

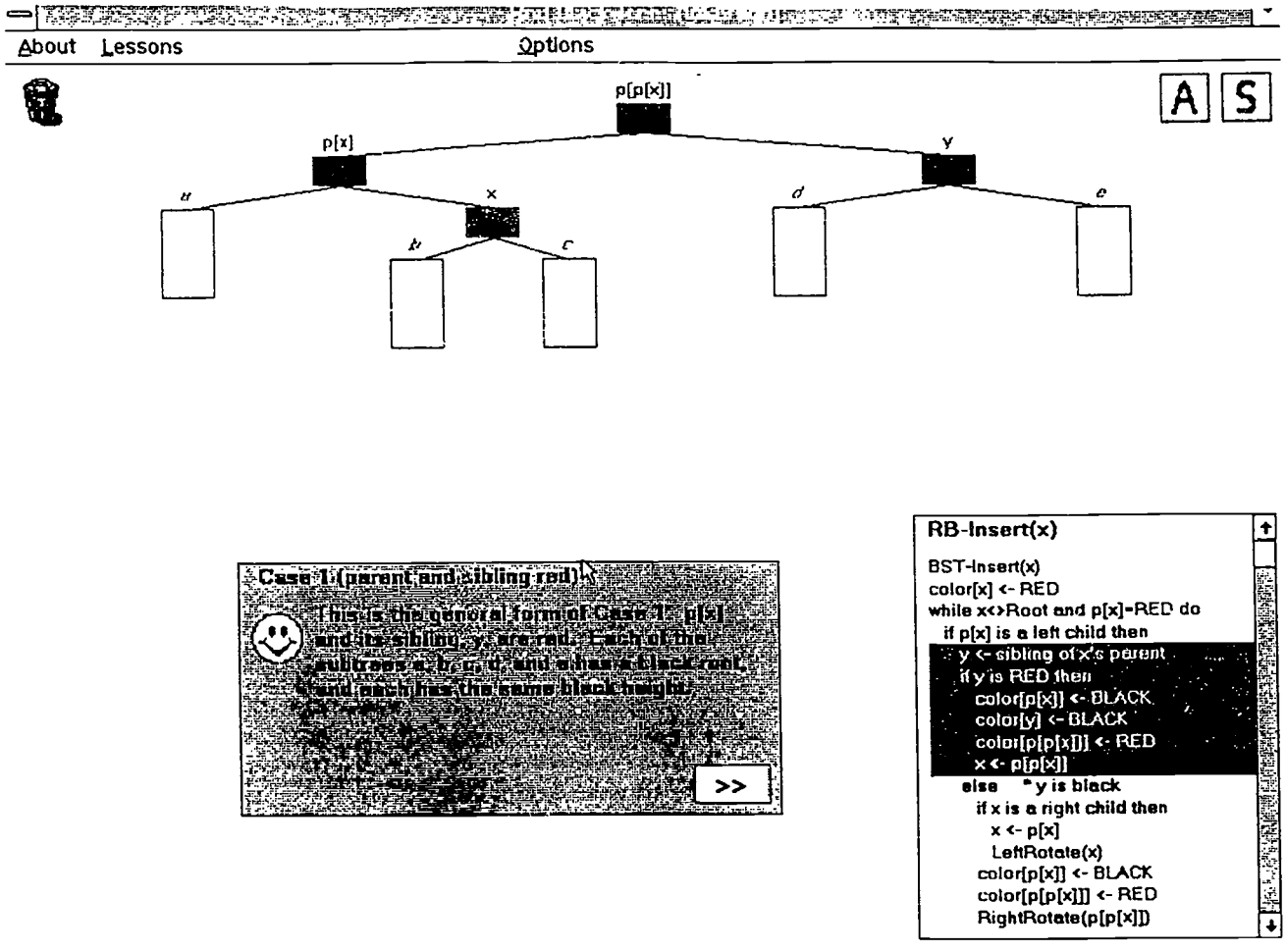

Figure 2. Canonical Case of Red-Black Trees.

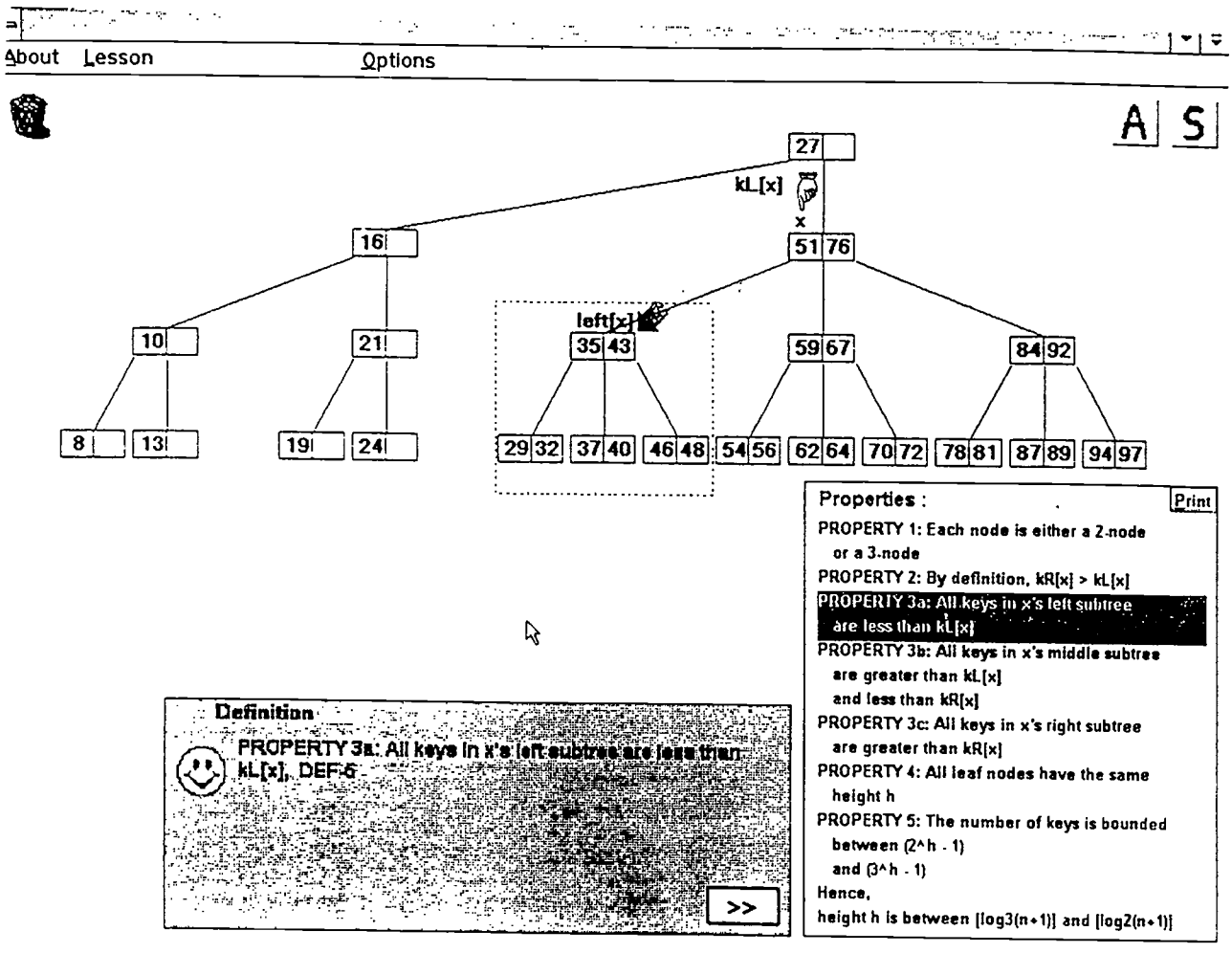

Figure 3. Example of a 2-3 Tree.